

Figure 1

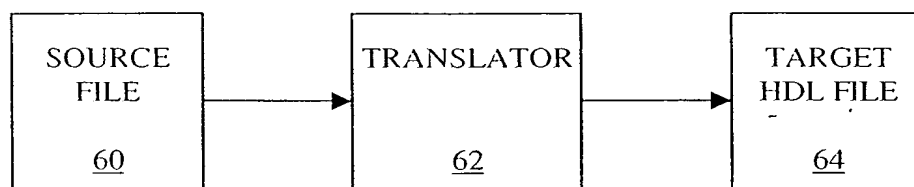


Figure 2

```

-----
-- File Name : complex_counter_v1.vhd
-- Update to : 21-Jan-2004
-- Designer  : Yinon Levy @ Royal Design
-----

```

```

library ieee ;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;
use work.Rx_package.all;

```

```

10 entity complex_counter_v1 is port

```

```

(
    ck          : in  std_logic          ; -- System clock
    mrstn       : in  std_logic          ; -- Asynchronous reset
    en1         : in  std_logic          ; -- Enable
    count       : out std_logic_vector(5 downto 0) -- Counter value
);
end complex_counter_v1 ;

```

```

architecture hdm of complex_counter_v1 is

```

```

    signal p_count :std_logic_vector(5 downto 0) ;

```

```

begin

```

```

    count <= p_count ;

```

```

-----
rx_complex_counter: process (ck, mrstn)

```

```

12 procedure reset_procedure is
14 begin

```

```

    p_count <= (others=> '0') ;
end reset_procedure ;

```

```

begin

```

```

16 RxProperties (ClkName=>"ck", ClkPolarity=> High,
18              RstName=> "mrstn", RstPolarity=> Low,
              SyncMode=>False, RstProc=>"reset_procedure");

```

```

----- Rx Statement -----
RxStatement (StatementName=> "count_statement", HitEventMode=> Serial) ;

```

```

20 HitEvent (StatementName=>"count_statement",
22           HitEventName=>"count_hit_event") ;

```

```

22 Trigger (HitEventName=>"count_hit_event",
24           TriggerExp=>% ##1 en1='1' ; ## 2 % );

```

```

24 RollingAction (HitEventName=>"count_hit_event",
26                TimeValueAction=> % ##12 p_count <= p_count + 1 %);

```

```

end process rx_complex_counter ;
end hdm ;

```

Figure 3

```

-----
-- File Name : complex_counter_v1.vhd
-- Update to : 21-Jan-2004
-- Designer  : Yinon Levy @ Royal Design
-----

```

```

library ieee ;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;
--use work.Rx_package.all;

```

```

40 → entity complex_counter_v1 is port
(
    ck          : in  std_logic          ; -- System clock
    mrstn       : in  std_logic          ; -- Asynchronous reset
    en1         : in  std_logic          ; -- Enable
    count       : out std_logic_vector(5 downto 0) -- Counter value
);
end complex_counter_v1 ;

architecture hdm of complex_counter_v1 is
    signal p_count :std_logic_vector(5 downto 0) ;

begin

    count <= p_count ;

```

```

-----
P_rx_complex_counter: process (ck, mrstn)

```

```

    procedure reset_procedure is
    begin
        p_count <= (others=> '0') ;
    end reset_procedure ;

```

```

    -- Handshake Rx-defined signals
    variable rd_counter_count_hit_event_0 : unsigned(3 downto 0) ;
    variable rd_end_trig_count_statement : std_logic ;
    variable rd_end_exec_count_statement : std_logic ;

```

```

-----
-- Rx Reset Procedures
-----

```

```

procedure rd_rst_count_statement is
begin
    rd_end_trig_count_statement := '0';
    rd_end_exec_count_statement := '0';
    rd_counter_count_hit_event_0 := (others=> '0');
end rd_rst_count_statement;

```

```

-----
-- Statement Name: count_statement
-- Statement Mode: Parallel
-----

```

Figure 4a

```

-----
-- Hit Event Name:      count_hit_event
-- Trigger Expression: ##1  en1='1'    ## 2
-----

102  procedure count_hit_event_evaluate  is
begin
    if (rd_counter_count_hit_event_0=0) then
        if (en1='1') then
            rd_counter_count_hit_event_0 := rd_counter_count_hit_event_0 + 1;
        end if;
    elsif (rd_counter_count_hit_event_0<=2) then
        rd_counter_count_hit_event_0 := rd_counter_count_hit_event_0 + 1;

        if (rd_counter_count_hit_event_0=3) then
            rd_counter_count_hit_event_0 := (others=> '0');
            rd_end_trig_count_statement := '1';
        end if;
    end if;
end count_hit_event_evaluate;

104  procedure count_hit_event_execute  is
begin
    if (rd_counter_count_hit_event_0<=11) then
        rd_counter_count_hit_event_0 := rd_counter_count_hit_event_0 + 1;
        p_count <= p_count+1;
    end if;
    if (rd_counter_count_hit_event_0=12) then
        rd_end_exec_count_statement := '1';
    end if;
end count_hit_event_execute;

106  procedure count_hit_event_activate  is
begin
    if (rd_end_trig_count_statement = '0') then
        count_hit_event_evaluate;
    end if;
    if (rd_end_trig_count_statement = '1'
        and rd_end_exec_count_statement = '0') then
        count_hit_event_execute;
    end if;
    if (rd_end_exec_count_statement = '1') then
        rd_rst_count_statement;
    end if;
end count_hit_event_activate;

-----
-- Rx Statements Activation Process
-----

108  begin
    if (mrstn = '0') then
        reset_procedure;
        rd_rst_count_statement;
    elsif (rising_edge(ck)) then
        count_hit_event_activate;
    end if;
end process;
end hdm ;

```

Figure 4b

Clock Polarity:

Positive ☒

Negative ☐

Figure 5

Memory Controller  
Pattern Recognizer  
Waveform Generator

Figure 6

Device Type:

Device Name:

Figure 7

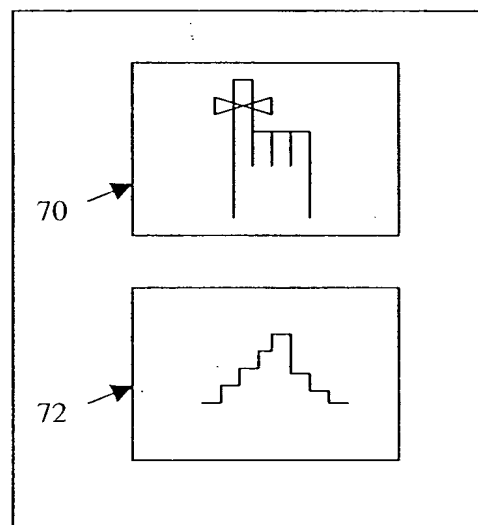


Figure 8

Sine  
Sawtooth

Figure 9

Amplitude:

Figure 10

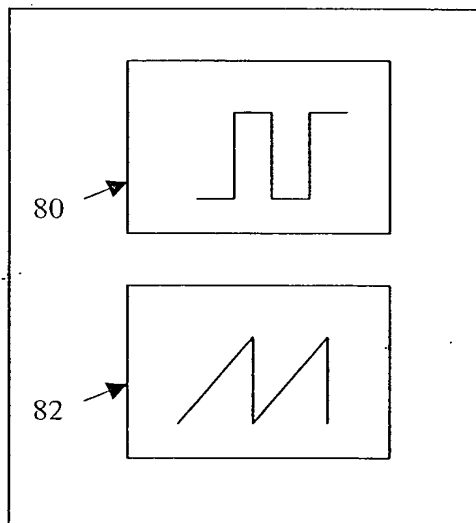


Figure 11

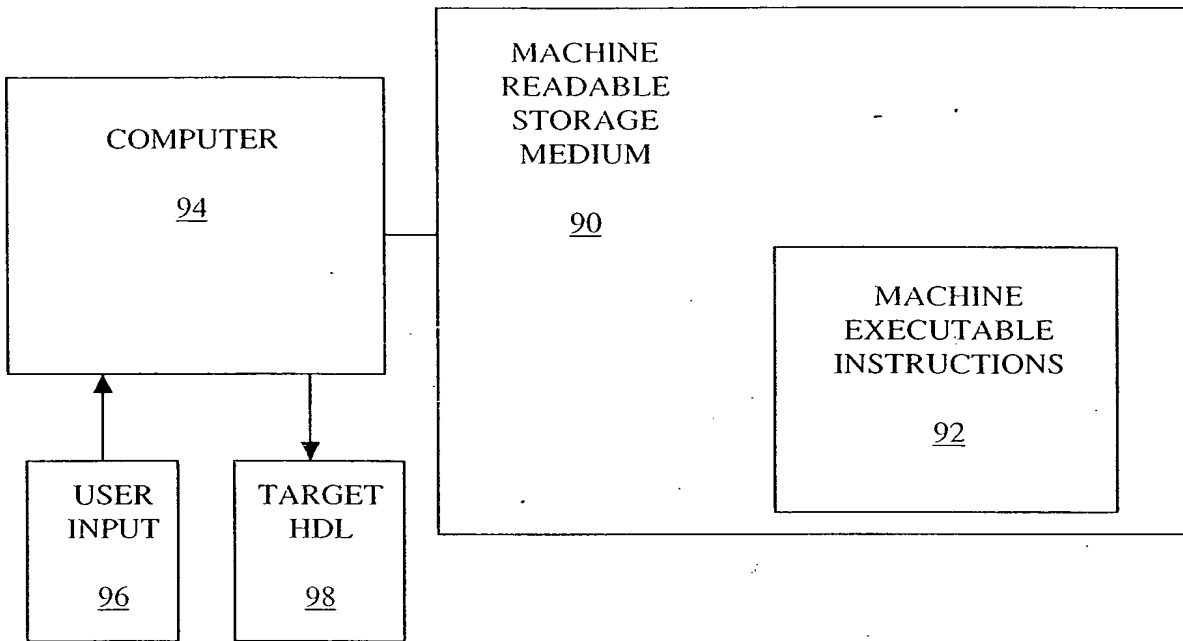


Figure 12